

# Cours 5 : Instructions de contrôle

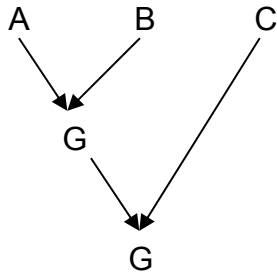
## Exercices

### Exercice 1

**Problème :** Calculer le plus grand de 3 nombres

**Méthode :**

- comparer les deux premiers nombres et définir le plus grand des deux
- comparer ce nombre et le troisième nombre



**Glossaire :**

- les trois nombres A, B et C
- le plus grand des trois nombres G

**Algorithme :**

début

lire les trois nombres A, B et C ;

si A est plus grand que B alors G est égal à A

sinon G est égal à B ;

si C est plus grand que G alors G est égal à C ;

fin

## Programme Matlab :

**%Calculer le plus grand de trois nombres**

```
T = input('Entrer les trois nombres :','s') ;
T = str2num(T) ;
A = T(1) ;
B = T(2) ;
C = T(3) ;
if A>B                                %comparer les deux premiers nombres
    G = A ;
else
    G = B ;
end
if C>G                                %comparer avec le troisième nombre
    G = C ;
end
disp(['Le plus grand nombre est : ',num2str(G)]) ;
```

### Alternative 1 :

**%Calculer le plus grand de trois nombres**

```
A = input('Entrer le premier nombre :');
B = input('Entrer le deuxième nombre :');
C = input('Entrer le troisième nombre :');
if A>B                                %comparer les deux premiers nombres
    G = A ;
else
    G = B ;
end
if C>G                                %comparer avec le troisième nombre
    G = C ;
end
disp(['Le plus grand nombre est : ',num2str(G)]);
```

### Alternative 2 :

**%Calculer le plus grand de trois nombres**

```
T = input('Entrer les trois nombres :','s') ;
T = str2num(T) ;
if T(1)>T(2)                            %comparer les deux premiers nombres
    G = T(1) ;
else
    G = T(2) ;
end
if T(3)>G                                %comparer avec le troisième nombre
    G = T(3) ;
end
disp(['Le plus grand nombre est : ',num2str(G)]);
```

### Alternative 3 :

```
%Calculer le plus grand de trois nombres

T = input('Entrer les trois nombres :','s') ;
T = str2num(T) ;
G = T(1) ;                                %initialiser G au premier nombre
for i=2 :3                                  %comparer G avec chaque nombre suivant
    if T(i)>G
        G = T(i) ;
    end
end
disp(['Le plus grand nombre est : ',num2str(G)]);
```

### Exercice 2 :

**Problème :** trouver le plus grand élément dans un vecteur de nombres

#### Méthode :

- définir le plus grand élément comme étant le premier élément du vecteur
- comparer cette valeur avec le deuxième élément du vecteur et définir le plus grand des deux
- comparer cette valeur avec le troisième élément du vecteur et définir le plus grand des deux
- faire de même jusqu'au dernier élément du vecteur

#### Glossaire :

- un vecteur de nombres X
- le nombre maximum MAX
- la longueur du vecteur TAILLE
- un compteur de nombres i

#### Algorithme :

```
début
lire le vecteur X ;
calculer la longueur du vecteur X ;
initialiser MAX au premier élément ;
de i = 2 à TAILLE faire
    si X(i)>MAX alors MAX = X(i)
écrire MAX ;
fin
```

## Programme Matlab :

*%Calculer le plus grand élément d'un vecteur de nombres*

```
X = input('Entrer le vecteur de nombres :','s');
X = str2num(X);
TAILLE = length(X);
MAX = X(1);                                %initialiser MAX au premier élément
for i=2 :TAILLE
    if X(i)>MAX                             %comparer MAX avec chaque élément suivant
        MAX = X(i) ;
    end
end
disp(['Le plus grand élément du vecteur est : ',num2str(MAX)]) ;
```

## Exercice 3 :

**Problème :** calculer le plus petit élément d'un vecteur de nombres

**Méthode :**

- définir le plus petit élément comme étant le premier élément du vecteur
- comparer cette valeur avec le deuxième élément du vecteur et définir le plus petit des deux
- comparer cette valeur avec le troisième élément du vecteur et définir le plus petit des deux
- faire de même jusqu'au dernier élément du vecteur

**Glossaire :**

- un vecteur de nombres X
- le nombre minimum MIN
- la longueur du vecteur TAILLE
- un compteur de nombres i

**Algorithme :**

```
début
lire le vecteur X ;
calculer la longueur du vecteur X ;
initialiser MIN au premier élément ;
de i = 2 à TAILLE faire
    si X(i)<MIN alors MIN = X(i)
écrire MIN ;
fin
```

## Programme Matlab:

*%Calculer le plus petit élément d'un vecteur de nombres*

X = input('Entrer le vecteur de nombres :','s');

X = str2num(X);

TAILLE = length(X);

MIN = X(1);

*%initialiser MIN au premier élément*

for i=2 :TAILLE

if X(i)<MIN

*%comparer MIN avec chaque élément suivant*

MIN = X(i) ;

end

end

disp(['Le plus petit élément du vecteur est : ',num2str(MIN)]);

## Exercice 4 :

**Problème :** faire la somme des éléments d'un vecteur

**Méthode :**

- sommer les éléments du vecteur X un à un jusqu'à la fin du vecteur

**Glossaire :**

- le vecteur de nombres : X
- la taille du vecteur X : TAILLE
- la somme des nombres : SOM

**Algorithme :**

début

lire le vecteur de nombres X ;

calculer la taille du vecteur TAILLE ;

initialiser la somme des nombres SOM ;

de i=1 à TAILLE faire

ajouter le  $i^{\text{ème}}$  nombre à SOM ;

écrire la somme ;

fin

## Programme Matlab :

```
%Somme des éléments d'un vecteur de nombres

X = input('Entrer le vecteur de nombres :','s');
X = str2num(X);
TAILLE = length(X);
if TAILLE==0                                %si aucun élément rentré, afficher un message
    disp('pas d"éléments dans le vecteur');
else
    SOM = 0;                                %initialiser la somme à 0
    for i=1:TAILLE                            %sommer un à un les éléments du vecteur
        SOM = SOM+X(i);
    end
    disp(['la somme des éléments du vecteur est :',num2str(SOM)]);
end
```

## Exercice 5 :

**Problème :** faire la somme des éléments des colonnes d'une matrice

**Méthode :**

- pour chaque colonne de la matrice, sommer les éléments

**Glossaire :**

- la matrice de nombres : A
- le nombre de lignes de la matrice A : NBL
- le nombre de colonnes de la matrice A : NBCOL
- le vecteur contenant la somme de chaque colonne : SOM

**Algorithme :**

```
début
lire la matrice de nombres A ;
calculer le nombre de lignes de la matrice NBL ;
calculer le nombre de colonnes de la matrice NBCOL ;
tant qu' on a une colonne faire
    initialiser la somme de la colonne ;
    tant qu' on a une ligne
        faire la somme des éléments
    écrire la somme des éléments des colonnes
fin
```

## Programme Matlab :

%Somme des éléments des colonnes d'une matrice

```
A = input('Entrer la matrice de nombres :','s');
A = str2num(A);
TAILLE = size(A);
NBL = TAILLE(1) ;
NBCOL = TAILLE(2);
for j=1:NBCOL                                %pour chaque colonne
    SOM(j) = 0;                               %initialiser la somme à 0
    for i=1:NBL                               %sommer un à un les éléments de la colonne
        SOM(j) = SOM(j)+A(i,j);
    end
end
disp(['la somme des éléments par colonne est :',num2str(SOM)]);
```

### Alternative1 :

%Somme des éléments des colonnes d'une matrice

```
A = input('Entrer la matrice de nombres :','s');
A = str2num(A);
TAILLE = size(A);
NBL = TAILLE(1) ;
NBCOL = TAILLE(2);
SOM=zeros(1,NBCOL);                         %initialiser le vecteur de somme à 0
for j=1:NBCOL                                %pour chaque colonne
    for i=1:NBL
        SOM(j) = SOM(j)+A(i,j);             %sommer un à un les éléments de la colonne
    end
end
disp(['la somme des éléments par colonne est :',num2str(SOM)]);
```

### Alternative2 :

%Somme des éléments des colonnes d'une matrice

```
A = input('Entrer la matrice de nombres :','s');
A = str2num(A);
TAILLE = size(A);
NBL = TAILLE(1) ;
NBCOL = TAILLE(2);
SOM=A(1, :);                                %initialiser le vecteur de somme à la première ligne
                                                %de la matrice
for j=1:NBCOL                                %pour chaque colonne
    for i=2:NBL                               %sommer un à un les éléments suivants
        SOM(j) = SOM(j)+A(i,j);             %de la colonne
    end
end
disp(['la somme des éléments par colonne est : ',num2str(SOM)]);
```

## Exercice 6 :

**Problème :** faire la moyenne des éléments d'un vecteur

**Méthode :**

- calculer la somme des éléments du vecteur
- calculer la moyenne des éléments du vecteur

**Glossaire :**

- le vecteur de nombres : X
- la taille du vecteur X : TAILLE
- la somme des nombres : SOM
- la moyenne des nombres : MOY

**Algorithme :**

```
début
lire le vecteur de nombres X ;
calculer la taille du vecteur TAILLE ;
initialiser la somme des nombres SOM ;
pour i=1 à TAILLE faire
    ajouter le ième nombre à SOM ;
calculer la moyenne MOY = SOM/TAILLE ;
écrire la somme ;
fin
```

**Programme Matlab :**

```
%Moyenne des éléments d'un vecteur

X = input('Entrer le vecteur de nombres :','s');
X = str2num(X);
TAILLE = length(X) ;
SOM = 0 ;                                %faire la somme des éléments du vecteur
for i=1:TAILLE
    SOM = SOM+X(i);
end
MOY = SOM/TAILLE ;                       %calculer la moyenne
disp(['la moyenne des éléments du vecteur est : ',num2str(MOY)]);
```



## Exercice 7 :

**Problème :** sur un ensemble de notes entre 0 et 20

- calculer le nombre de notes supérieures ou égales à 10
- calculer le nombre de notes supérieures à la moyenne des notes

**Méthode :**

- calculer la moyenne des notes
- comparer chaque note avec 10 et la moyenne des notes
- compter le nombre de notes supérieures à 10 et le nombre de notes supérieures à la moyenne

**Glossaire :**

- le vecteur de notes : NOTE
- le nombre de notes : TAILLE
- la somme des notes : SOM
- la moyenne des notes : MOY
- le compteur de notes supérieures ou égales à 10 : C10
- le compteur de notes supérieures à MOY : CMOY
- le compteur de notes dans le vecteur : i

**Algorithme :**

début

lire le vecteur de notes NOTE;

calculer le nombre de notes TAILLE (la longueur du vecteur) ;

initialiser la somme des notes SOM à 0 ;

**%Calculer la somme des notes SOM ;**

tant qu'on n'est pas arrivé à la fin du vecteur de notes faire la somme des notes ;

calculer la moyenne des notes ;

initialiser les compteurs C10 et CMOY à 0 ;

tant que on n'est pas arrivé à la fin du vecteur de notes faire

    si la note i est supérieure ou égale à 10 alors incrémenter le compteur C10 ;

    si la note est supérieure ou égale à MOY alors incrémenter le compteur CMOY

afficher C10 et CMOY ;

fin

## Programme Matlab :

```
%Calculer le nombre de notes supérieures ou égales à 10 et le nombre de notes
%supérieures ou égales à la moyenne

NOTE = input('Entrer le vecteur de nombres :','s');
NOTE = str2num(NOTE);
TAILLE = length(NOTE);
SOM = 0 ;                                %calculer la somme et la moyenne des notes
for i=1:TAILLE
    SOM = SOM+NOTE(i) ;
end
MOY = SOM/TAILLE ;
C10 = 0 ;                                %initialiser les compteurs de notes à 0
CMOY = 0 ;
for i=1 :TAILLE                          %pour chaque note
    if NOTE(i)>=10                        %comptabiliser la note si elle est supérieure à 10
        C10 = C10+1 ;
    end
    if NOTE(i)>=MOY                      %comptabiliser la note si elle est supérieure à
        CMOY = CMOY+1 ;                %la moyenne des notes
    end
end
disp(['Nombre de notes supérieures à 10 : ',num2str(C10)]) ;
disp(['Nombre de notes supérieures à la moyenne ',num2str(MOY),' : ', num2str(CMOY)]) ;
```

**Exemple :** randint(1,100,[1,20]) génère un vecteur de 100 notes entre 1 et 20

## Exercice 8 :

**Problème :** compter le nombre de mots de 2 lettres dans un texte

### Méthode :

- lire chaque mot du texte : on définit les mots comme étant séparés par des caractères de type espace
- compter le nombre de lettres de chaque mot
- compter le nombre de mots de 2 lettres

### Glossaire :

- la chaîne de caractères : s
- la taille du texte : TAILLE
- le compteur de mots de 2 lettres : NBMOT
- le compteur de lettres dans un mot : L
- le compteur de caractères lus : i

## Algorithme :

```
début
lire le texte ;
calculer la taille du texte ;
initialiser la nombre de mot NBMOT ;
initialiser le compteur de caractères lus ;
%Boucle 1 : lecture du texte
tant que on n'est pas arrivé à la fin du texte faire
    initialiser le compteur de lettres L ;
    %Boucle2 : lecture d'un mot
    %tant qu'on n'a pas un caractère de type espace
    tant on n'a pas fini de lire un mot faire
        incrémenter le compteur de lettres L ;
        lire le caractère suivant ;
        %sortir de la boucle si on a atteint la fin du texte i.e. le compteur de
        %caractères i > TAILLE
        si on est arrivé à la fin du texte alors sortir de la boucle ;
    %fin de la boucle 2
    si le nombre de lettres est égale à 2 alors
        incrémenter le nombre de mots NBMOT ;
    sinon
        lire le caractère suivant ;
    %fin de la boucle 1
écrire le nombre de mots de 2 lettres ;
fin
```

## Programme Matlab :

```
%Compter le nombre de mots de 2 lettres dans un texte

s = input('Entrer le texte :','s');
TAILLE = length(s) ;
delimiter = [9:13 32];
NBMOT = 0 ;
i =1;
while i<=TAILLE
    L = 0;
    while (~any(s(i)==delimiter))
        L=L+1;
        i=i+1;
        if i>TAILLE
            break
        end
    end
    if L==2
        NBMOT = NBMOT+1;
    end
    i=i+1 ;
end
disp(['il y a ',num2str(NBMOT),' mots de deux lettres dans le texte']);

%définir tous les caractères de type espace
%initialiser le nombre de mots de 2 lettres

%Boucle 1 : tant qu'on n'est pas à la fin du texte
%initialiser le compteur de lettres
%Boucle 2 : tant que le caractère lu n'est pas
%un espace
%incrémenter le nombre de lettres
%passer au caractère suivant
%si on a atteint la fin du texte
%alors sortir de la boucle 2

%fin de la boucle 2
%comptabiliser le mot s'il contient 2 lettres

%passer au caractère suivant
%fin de la boucle 1
```

## Alternative1 :

%Compter le nombre de mots de 2 lettres dans un texte

```
s = input('Entrer le texte :','s');
TAILLE = length(s) ;
delimiter = [9:13 32];
NBMOT = 0 ;
i =1;
while i<=TAILLE
    L = 0;
    while (~any(s(i)==delimiter))
        L=L+1;
        if i==TAILLE
            break
        end
        i=i+1;
    end
    if L==2
        NBMOT = NBMOT+1;
    end
    i=i+1 ;
end
disp(['il y a ',num2str(NBMOT),' mots de deux lettres dans le texte']);
```

%définir tous les caractères de type espace  
%initialiser le nombre de mots de 2 lettres  
%tant qu'on n'est pas à la fin du texte  
%initialiser le compteur de lettres  
%tant que le caractère lu n'est pas un espace  
%incrémenter le nombre de lettres  
%si on a atteint la fin du texte, sortir de la boucle  
%passer au caractère suivant  
%comptabiliser le mot s'il contient 2 lettres  
%passer au caractère suivant

**Exemple :** 'il faut résoudre ce problème'

**Remarque :** choix d'une première boucle **WHILE** au lieu d'une boucle **FOR**

On pourrait être tenté d'utiliser l'instruction **FOR** pour décrire la première boucle comme suit :

%Compter le nombre de mots de 2 lettres dans un texte

```
s = input('Entrer le texte :','s');
TAILLE = length(s) ;
delimiter = [9:13 32];
NBMOT = 0 ;
for i=1:TAILLE
    L = 0;
    while (~any(s(i)==delimiter))
        L=L+1;
        if i==TAILLE
            break
        end
        i=i+1;
    end
    if L==2
        NBMOT = NBMOT+1;
    end
end
disp(['il y a ',num2str(NBMOT),' mots de deux lettres dans le texte']);
```

%définir tous les caractères de type espace  
%initialiser le nombre de mots de 2 lettres  
%tant qu'on n'est pas à la fin du texte  
%initialiser le compteur de lettres  
%tant que le caractère lu n'est pas un espace  
%incrémenter le nombre de lettres  
%si on a atteint la fin du texte, sortir de la boucle  
%passer au caractère suivant  
%comptabiliser le mot s'il contient 2 lettres

Ce type de programme ne permet pas d'atteindre la solution du problème pour la raison suivante. Dans la deuxième boucle (lecture d'un mot), on est amené à modifier la valeur du compteur d'indices *i* utilisé par la première boucle. Cependant, l'utilisation de l'instruction FOR ne retient pas la valeur du compteur *i* modifiée dans la boucle 2. Elle incrémentera la valeur de *i* non pas relativement à la nouvelle valeur de *i* mais plutôt relativement à son ancienne valeur dans la boucle 1. Dès lors, les opérations voulues ne se feront pas sur la valeur de *i* modifiée dans la boucle 2 ainsi que le programme l'exige pour atteindre la solution souhaitée.

**Analyse du programme** : pour le texte « ce programme ne marche pas »

TAILLE = 26  
NBMOT = 0

**Boucle 1 :**

**Etape 1 : i=1**

initialiser L : L=0

**Boucle 2 :**

**Etape 1 : s(1)= « c » n'est pas un espace**

incrémenter L : L=1

i=1 est différent de TAILLE=26 : ne pas sortir de la boucle 2

incrémenter i : i=2

**Etape 2 : s(2)= « e » n'est pas un espace**

incrémenter L : L=2

i=2 est différent de TAILLE=26 : ne pas sortir de la boucle 2

incrémenter i : i=3

**Etape 3 : s(3)= « » est un espace**

Sortir de la boucle 2

évaluer L : L=2

Incrémenter NBMOT : NBMOT=1

passer à l'étape 2 de la boucle 1

**Etape 2 : i=2**

Le problème se situe à cette étape : on souhaite évaluer le mot suivant, c'est-à-dire considérer la première lettre du mot « programme » située à la position i=4.

Au lieu d'incrémenter la nouvelle valeur de *i* (i=3), l'instruction FOR va incrémenter l'ancienne valeur du compteur d'indices *i* évaluée au début de l'étape 1 de la Boucle 1 (soit i=1). Cela revient à considérer la lettre située à la position i=2 (soit la lettre « e ») au lieu de la première lettre du mot suivant (soit la lettre « p ») !

## **Exercice 9 : p.32 du cours (CONTINUE)**

**Problème** : compter le nombre de lignes dans un fichier (excepté les commentaires)

**Méthode** :

- lire chaque ligne du fichier
- compter cette ligne uniquement si elle n'est pas vide ou si ce n'est pas une ligne de commentaires

## Glossaire :

- l'identificateur de fichier à l'ouverture : fid
- le compteur de lignes : count
- un ligne du fichier : line

## Algorithme :

début

ouvrir le fichier en lecture ;

initialiser le compteur de lignes ;

tant que on n'est pas arrivé à la fin du fichier faire

    lire une ligne ;

    si la ligne est vide ou est un commentaire alors

        la ligne n'est pas comptée et on passe à la ligne suivante ;

    incrémenter le compteur de lignes ;

fermer le fichier ;

écrire le nombre de lignes du fichier ;

fin

## Algorithme alternatif :

début

ouvrir le fichier en lecture ;

initialiser le compteur de lignes ;

tant que on n'est pas arrivé à la fin du fichier faire

    lire une ligne ;

    si la ligne n'est pas vide et n'est pas un commentaire alors

        incrémenter le compteur de lignes ;

fermer le fichier ;

écrire le nombre de lignes du fichier ;

fin

## Programme Matlab :

**% Compter le nombre de lignes dans un fichier (excepté les commentaires)**

fid=fopen('spm.m','r');

**%ouvrir le fichier en lecture**

count = 0;

**%initialiser le compteur de lignes**

while ~feof(fid)

**%tant qu'on n'est pas à la fin du fichier**

    line = fgetl(fid);

**%lire une ligne**

    if isempty(line) | strncmp(line, '%', 1)

**%si elle est vide ou est un commentaire**

        continue

**%passer à l'itération suivante de la boucle**

    end

    count = count+1;

**%compter la ligne**

end

disp(sprintf('%d lignes', count));

fclose(fid);

**%fermer le fichier**

**Remarque :** l'utilisation de la commande *fgetl* permet de renvoyer dans la variable *line* le contenu d'une ligne.

A l'ouverture du fichier, on se trouve au début du fichier ce qui aura pour effet de lire la première ligne du fichier. La lecture de la ligne par *fgetl* aura pour effet de déplacer le pointeur de position du fichier au début de la ligne suivante.

Lors de l'itération suivante de la boucle, on lira alors la ligne suivante.

#### Alternative :

```
% Compter le nombre de lignes dans un fichier (excepté les commentaires)

fid=fopen('spm.m','r');           %ouvrir le fichier en lecture
count = 0;                        %initialiser le compteur de lignes
while ~feof(fid)                  %tant qu'on n'est pas à la fin du fichier
    line = fgetl(fid);             %lire une ligne
    if ~isempty(line) & ~strncmp(line, '%', 1) %si elle n'est pas vide
                                    %ou n'est pas un commentaire
        count=count+1;            %compter la ligne
    end
end

disp(sprintf('%d lignes', count));
fclose(fid);                       %fermer le fichier
```

**Exemple :** compter le nombre de lignes du fichier spm.m

### Exercice 10 : p.34 du cours (BREAK)

**Problème :** afficher les lignes de commentaires en entête d'un fichier

#### Méthode :

- lire et afficher chaque ligne du fichier jusqu'à ce qu'on rencontre la première ligne vide (fin des commentaires en entête du fichier)

#### Glossaire :

- l'identificateur de fichier à l'ouverture : fid
- une ligne du fichier : line

#### Algorithme :

```
début
ouvrir le fichier en lecture ;
tant que on n'est pas arrivé à la fin du fichier faire
    lire une ligne ;
    si la ligne est vide alors ne plus lire de lignes dans le fichier
    afficher la ligne
fermer le fichier
fin
```

## Algorithme alternatif :

```
début  
ouvrir le fichier en lecture ;  
tant que on n'est pas arrivé à la fin du fichier faire  
    lire une ligne ;  
    si la ligne n'est pas vide alors afficher la ligne  
    sinon ne plus lire de lignes dans le fichier  
fermer le fichier  
fin
```

## Programme Matlab :

```
%Afficher les lignes de commentaires en entête d'un fichier  
  
fid=fopen('spm.m','r');           %ouvrir le fichier en lecture  
while ~feof(fid)                 %tant qu'on n'est pas à la fin du fichier  
    line = fgetl(fid);            %lire une ligne  
    if isempty(line)              %si elle est vide  
        break                    %alors sortir de la boucle  
    end  
    disp(line)                   %afficher la ligne de commentaires  
end  
fclose(fid);                     %fermer le fichier
```

## Alternative :

```
%Afficher les lignes de commentaires en entête d'un fichier  
  
fid=fopen('spm.m','r');           %ouvrir le fichier en lecture  
while ~feof(fid)                 %tant qu'on n'est pas à la fin du fichier  
    line = fgetl(fid);            %lire une ligne  
    if ~isempty(line)             %si elle n'est pas vide  
        disp(line)               %alors afficher la ligne de commentaires  
    else break;                  %sinon sortir de la boucle  
    end  
end  
fclose(fid);                     %fermer le fichier
```

## Exercice 11 : p.36 du cours (RETURN)

**Problème :** rechercher la première occurrence d'une lettre dans une chaîne de caractères et retourner sa position dans le texte

### Méthode :

- lire le texte jusqu'à ce que la lettre soit trouvée ou jusqu'à la fin du texte
- arrêter l'exécution du programme si le texte ou la lettre sont vides



## Glossaire :

- le texte : texte
- la lettre à chercher dans le texte : lettre
- la longueur du texte : lent
- la longueur de la lettre : lenl
- le compteur de position de la lettre dans le texte : i

## Algorithme :

début

lire le texte ;

lire la lettre à trouver dans le texte ;

calculer la longueur du texte ;

calculer la longueur de la lettre ;

si le texte est vide ou si la lettre est vide alors

    afficher un message ;

    arrêter l'exécution du programme ;

tant que la lettre n'est pas trouvée dans le texte faire

    incrémenter le compteur de position ;

    si tout le texte a été lu alors

        arrêter l'exécution du programme ;

afficher la position de la lettre dans le texte ;

fin

## Programme Matlab :

**%Rechercher la première occurrence d'une lettre dans une chaîne de caractères et retourner sa position dans le texte**

texte=input('Entrer le texte : ','s');

lettre=input('Saisir la lettre : ','s');

lent = length(texte);

lenl = length(lettre);

if lenl == 0 || lent == 0

    disp('lettre ou chaine vide');

    return

**%si le texte ou la lettre est vide**

**%alors afficher un message**

**%sortir du programme**

end

i = 1;

while texte(i)~=lettre

    i = i+1;

    if i>lent

        return

    end

**%tant que la lettre n'est pas trouvée**

**%passer au caractère suivant**

**%si tout le texte a été parcouru**

**%alors sortir du programme**

end

disp(['la lettre ',texte(i),' se trouve à la position ',num2str(i)]);

### Alternative :

*%Rechercher la première occurrence d'une lettre dans une chaîne de caractères et retourner sa position dans le texte*

```
texte=input('Entrer le texte : ','s');
lettre=input('saisir la lettre : ','s');
lent = length(texte);
lenl = length(lettre);
if isempty(texte) || isempty(lettre)           %si le texte ou la lettre est vide
    disp('lettre ou chaine vide');             %alors afficher un message
    return                                     %sortir du programme
end
i = 1;
while texte(i)~=lettre                         %tant que la lettre n'est pas trouvée
    i = i+1;                                   %passer au caractère suivant
    if i>lent                                  %si tout le texte a été parcouru
        return                                %alors sortir du programme
    end
end
disp(['la lettre ',texte(i),' se trouve à la position ',num2str(i)]);
```

### **Exercice 12 : P.38 du cours (ERROR)**

**Problème :** reprendre le problème 11 et afficher un message d'erreur qui stoppe l'exécution du programme si le texte ou la lettre saisis sont vides

### **Méthode :**

- lire le texte jusqu'à ce que la lettre soit trouvée ou jusqu'à la fin du texte
- arrêter l'exécution du programme si le texte ou la lettre sont vides

### **Glossaire :**

- le texte : texte
- la lettre à chercher dans le texte : lettre
- la longueur du texte : lent
- la longueur de la lettre : lenl
- le compteur de position de la lettre dans le texte : i

## Algorithme :

```
début
lire le texte ;
calculer la longueur du texte ;
si le texte est vide alors
    afficher un message d'erreur qui stoppe l'exécution du programme
lire la lettre à trouver dans le texte ;
calculer la longueur de la lettre ;
si la lettre est vide alors
    afficher un message d'erreur qui stoppe l'exécution du programme
tant que la lettre n'est pas trouvée dans le texte faire
    incrémenter le compteur de position ;
    si tout le texte a été lu alors
        arrêter l'exécution du programme ;
afficher la position de la lettre dans le texte ;
fin
```

## Programme Matlab :

```
%Rechercher la première occurrence d'une lettre dans une chaîne de caractères et
retourner sa position dans le texte
%Afficher un message d'erreur qui stoppe l'exécution du programme si le texte ou la lettre
%saisis sont vides

texte=input('Entrer le texte : ','s');
lent = length(texte);
if lent == 0
    error('chaine vide')
end

lettre=input('saisir la lettre : ','s');
lenl = length(lettre);
if lenl == 0
    error('lettre vide')
end

i = 1;
while texte(i)~=lettre
    i = i+1;
    if i>lent
        return
    end
end
disp(['la lettre ',texte(i),' se trouve à la position ',num2str(i)]);
```

### Exercice 13 : p.40 du cours (WARNING)

**Problème :** reprendre le problème 11, afficher un message d'avertissement et demander à l'utilisateur de saisir le texte et la lettre jusqu'à ce qu'ils ne soient plus vides

#### Méthode :

- demander à l'utilisateur de saisir un texte et une lettre jusqu'à ce qu'ils ne soient plus vides
- lire le texte jusqu'à ce que la lettre soit trouvée ou jusqu'à la fin du texte

#### Glossaire :

- le texte : texte
- la lettre à chercher dans le texte : lettre
- la longueur du texte : lent
- la longueur de la lettre : lenl
- le compteur de position de la lettre dans le texte : i

#### Algorithme :

début

lire le texte ;

calculer la longueur du texte ;

tant le texte saisi est vide alors

    afficher un message d'avertissement ;

    lire le texte à nouveau ;

    calculer la longueur du texte ;

lire la lettre à trouver dans le texte ;

calculer la longueur de la lettre ;

tant la lettre saisie est vide alors

    afficher un message d'avertissement ;

    lire la lettre à nouveau ;

    calculer la longueur de la lettre ;

tant que la lettre n'est pas trouvée dans le texte faire

    incrémenter le compteur de position ;

    si tout le texte a été lu alors

        arrêter l'exécution du programme ;

afficher la position de la lettre dans le texte ;

fin

## Programme Matlab :

%Rechercher la première occurrence d'une lettre dans une chaîne de caractères et retourner sa position dans le texte  
%Afficher un message d'avertissement et demander à l'utilisateur de saisir le texte et la lettre jusqu'à ce qu'ils ne soient plus vides

```
texte=input('Entrer le texte : ','s');
lent = length(texte);
while lent == 0                                %tant que le texte est vide
    warning('chaine vide');                    %afficher un message d'avertissement
    texte=input('Entrer le texte : ','s');      %entrer le texte à nouveau
    lent = length(texte);
end

lettre=input('saisir la lettre : ','s');
lenl = length(lettre);
while lenl == 0                                %tant que la lettre est vide
    warning('lettre vide');                    %afficher un message d'avertissement
    lettre=input('Entrer la lettre : ','s');    %entrer la lettre à nouveau
    lenl = length(lettre);
end

i = 1;
while texte(i)~=lettre                        %tant que la lettre n'est pas trouvée
    i = i+1;                                  %passer au caractère suivant
    if i>lent                                  %si tout le texte a été parcouru
        return                                %alors sortir du programme
    end
end
disp(['la lettre ',texte(i),' se trouve à la position ',num2str(i)]);
```

## Exercice 14 : p.42 du cours (PAUSE)

### Problème :

- reprendre le problème 11, afficher un message d'avertissement et demander à l'utilisateur de saisir le texte et la lettre jusqu'à ce qu'ils ne soient plus vides
- faire une pause de 5 secondes entre la saisie du texte et la saisie de la lettre
- demander à l'utilisateur de taper une touche du clavier pour afficher la position de la lettre

### Méthode :

- demander à l'utilisateur de saisir un texte jusqu'à ce qu'il ne soit plus vide
- faire une pause de 5 secondes
- demander à l'utilisateur de saisir une lettre jusqu'à ce qu'elle ne soit plus vide
- lire le texte jusqu'à ce que la lettre soit trouvée ou jusqu'à la fin du texte
- demander à l'utilisateur de taper une touche du clavier pour afficher la position de la lettre

## Glossaire :

- le texte : texte
- la lettre à chercher dans le texte : lettre
- la longueur du texte : lent
- la longueur de la lettre : lenl
- le compteur de position de la lettre dans le texte : i

## Algorithme :

### début

lire le texte ;

calculer la longueur du texte ;

tant le texte saisi est vide alors

    afficher un message d'avertissement ;

    lire le texte à nouveau ;

    calculer la longueur du texte ;

faire une pause de 5 secondes ;

lire la lettre à trouver dans le texte ;

calculer la longueur de la lettre ;

tant la lettre saisie est vide alors

    afficher un message d'avertissement ;

    lire la lettre à nouveau ;

    calculer la longueur de la lettre ;

tant que la lettre n'est pas trouvée dans le texte faire

    incrémenter le compteur de position ;

    si tout le texte a été lu alors

        arrêter l'exécution du programme ;

afficher un message ;

faire une pause jusqu'à ce qu'une touche du clavier soit tapée ;

afficher la position de la lettre dans le texte ;

### fin

## Programme Matlab :

```
%Rechercher la première occurrence d'une lettre dans une chaîne de caractères et
%retourner sa position dans le texte
%Afficher un message d'avertissement et demander à l'utilisateur de saisir le texte et la
%lettre jusqu'à ce qu'ils ne soient plus vides
%Faire une pause de 5 secondes entre la saisie du texte et la saisie de la lettre

texte=input('Entrer le texte : ','s');
lent = length(texte);
while lent == 0                                     %tant que le texte est vide
    warning('chaîne vide');                          %afficher un message d'avertissement
    texte=input('Entrer le texte : ','s');           %entrer le texte à nouveau
    lent = length(texte);
end
pause(5);                                           %faire une pause de 5 secondes
lettre=input('saisir la lettre : ','s');
lenl = length(lettre);
while lenl == 0                                     %tant que la lettre est vide
    warning('lettre vide');                          %afficher un message d'avertissement
    lettre=input('Entrer la lettre : ','s');          %entrer la lettre à nouveau
    lenl = length(lettre);
end

i = 1;
while texte(i)~=lettre                             %tant que la lettre n'est pas trouvée
    i = i+1;                                         %passer au caractère suivant
    if i>lent                                       %si tout le texte a été parcouru
        return                                     %alors sortir du programme
    end
end
disp('pour avoir le résultat, taper sur une touche du clavier !');
pause;                                             %faire une pause jusqu'à ce qu'une touche
                                                    %du clavier soit tapée
disp(['la lettre ',texte(i),' se trouve à la position ',num2str(i)]);
```